REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
05-10-2015	Final	26 Aug 2011 – 26 Aug 2015

4. TITLE AND SUBTITLE

Intelligence in the Now: Robust Intelligence in Complex Domains

5a. CONTRACT NUMBER FA2386-10-1-4135

5b. GRANT NUMBER Grant 104135

5c. PROGRAM ELEMENT NUMBER 61102F

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

6. AUTHOR(S)

Dr. Leslie Kaelbling

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Massachusetts Institute of Technology
32 Vassar Street, 32-G486
Cambridge 02139-4307
United States

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AOARD
UNIT 45002
APO AP 96338-5002

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/AFOSR/IOA(AOARD)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

104135

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Code A: Approved for public release, distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Our overall goal is to develop the estimation, planning, and control techniques necessary to enable robots to perform robustly and intelligently in complex uncertain domains. Robots operating in complex, unknown environments have to deal explicitly with uncertainty. Sensing is increasingly reliable, but inescapably local: robots cannot see, immediately, inside cupboards, under collapsed walls, or into nuclear containment vessels. Task planning, whether in household and disaster-relief domains, requires explicit consideration of uncertainty and the selection of actions at both the task and motion levels to support gathering information. Our approach to robust behavior in uncertain domains is founded on the notion of integrating estimation, planning, and execution in a feedback loop. A plan is made, based on the current belief state; the first step is executed; an observation is obtained; the belief state is updated; the plan is recomputed, if necessary, etc. We call this online replanning. Our work in this grant has developed an initial version of such a planner and demonstrated it for controlling the behavior of an autonomous mobile-manipulation robot.

15. SUBJECT TERMS

Artificial Intelligence, Automated Reasoning, Robotics

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Brian Lutz, Lt Col, USAF
U	U	U	SAR	28	**19b. TELEPHONE NUMBER** *(Include area code)* +81-42-511-2000

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Final Report for AOARD Grant FA2386-10-1-4135
Intelligence in the Now: Robust Intelligence in Complex Domains

September 26, 2015

Leslie Pack Kaelbling
lpk@csail.mit.edu
MIT Computer Science and Artificial Intelligence Laboratory
32 Vassar Street, Cambridge, MA 02139
Voice: 617-253-9695
Fax: 617-452-5034

Tomas Lozano-Perez
tlp@csail.mit.edu
MIT Computer Science and Artificial Intelligence Laboratory
32 Vassar Street, Cambridge, MA 02139
Voice: 617-253-7889
Fax: 617-452-5034

Period of Performance: 8/26/2011 - 8/26/2015

Abstract

Our overall goal is to develop the estimation, planning, and control techniques necessary to enable robots to perform robustly and intelligently in complex uncertain domains. Robots operating in complex, unknown environments have to deal explicitly with uncertainty. Sensing is increasingly reliable, but inescapably local: robots cannot see, immediately, inside cupboards, under collapsed walls, or into nuclear containment vessels. Task planning, whether in household and disaster-relief domains, requires explicit consideration of uncertainty and the selection of actions at both the task and motion levels to support gathering information.

In order to explicitly consider the effects of uncertainty and to generate actions that gain information, it is necessary to plan in *belief space*: that is, the space of the robot's beliefs about the state of its environment, which we will represent as probability distributions over states of the environment. For planning purposes, the initial state is a belief state and the goal is a set of belief states: for example, a goal might be for the robot to believe with probability greater than 0.99 that all of the groceries are put away in an acceptable location, or that there are no survivors remaining in the rubble.

Planning in belief space beautifully integrates perception and action, both of which affect beliefs in ways that can be modeled and thus exploited to achieve an ultimate goal. However, planning in belief space for realistic problems poses some substantial challenges: (a) belief space is generally a high-dimensional continuous space (of distributions) and (b) the outcomes of actions and (especially) perception makes the process dynamics highly non-deterministic. These are fundamental reasons why optimal planning under uncertainty is intractable in the worst case. But, approximate representation and planning algorithms are possible for many domains and belief space serves as an organizing principle in these approaches.

Our approach to robust behavior in uncertain domains is founded on the notion of integrating estimation, planning, and execution in a feedback loop. A plan is made, based on the current belief state; the first step is executed; an observation is obtained; the belief state is updated; the plan is recomputed, if necessary, etc. We call this *online replanning*. In contrast to the more typical method of finding a complete policy for all possible belief states in advance, this strategy allows planning to be efficient but approximate: it is important that the first step of the plan be useful, but the rest will be re-examined in light of the results of the first step.

A critical component of such a system is a planner that works effectively in very high-dimensional geometric problems that have substantial uncertainty: a robot trying to assemble ingredients for cooking a meal has to work in a space that is made up of the positions, orientations, and other aspects of a large number

of objects; it will have localized uncertainty about some of the objects and may have very little information about others. Planning for the robot is not just motion planning: it must decide what order to move objects in, how to grasp them, where to place them, and so on. It must also plan to gain information, including deciding where to look, determining that it must move objects out of the way to get an unoccluded view, or selecting a cupboard to search for a particular object it needs.

Our work in this grant has developed an initial version of such a planner and demonstrated it for controlling the behavior of an autonomous mobile-manipulation robot.

Introduction

Our work on this project attempts to integrate the key ideas in three broad areas of research within the Artificial Intelligence and Robotics community.

- Symbolic planning in the STRIPS tradition provides methods for dealing with a wide class of knowledge and goals. The methods in this area, derived from logic, provide powerful techniques for dealing with large problems, notably factoring and abstraction.
- Motion planning provides powerful methods for dealing with the geometric and kinematic constraints that are fundamental to robot motion.
- Decision-theoretic planning under uncertainty, in particular, formulations of problems as Partially Observable Markov Decision Problems (POMDP) (Figure 1), provides powerful methods for dealing with uncertainty and for integrating perception and action.

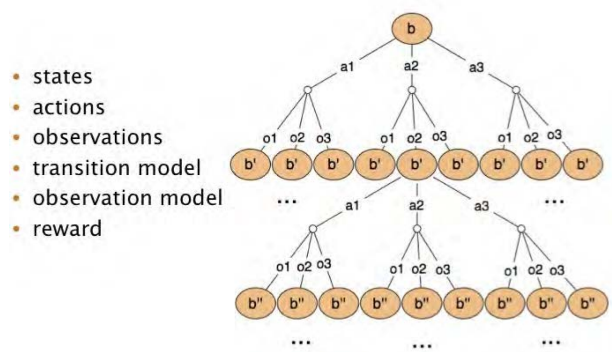


Figure 1: POMDP Model

These sub-fields have developed largely independently yet we believe that they each provide crucial components of the solution we seek.

When acting in an uncertain environment (Figure 2), a system should keep a characterization of its knowledge of the state of the world, the *belief*, as a probability distribution over the states; this is the job of the *state estimator* (SE). The *policy* (π) encodes the action to be taken for any belief. The central notion in our approach is that of planning in the *belief space* – the space of beliefs. The key issue is how to make planning in belief space tractable; traditional solvers for POMDPs, which construct complete belief-space policies through off-line computation are only useful for smallish problems.

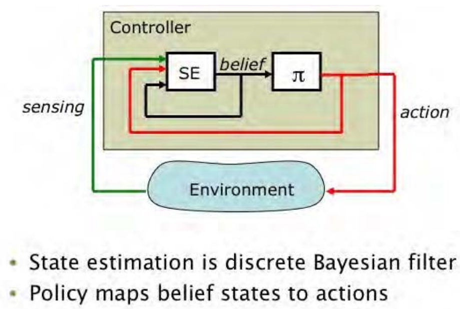


Figure 2: POMDP Controller

Experiment

The key objective of this research is to develop principled and practical approaches for robot decision-making under uncertainty. Our principal experimental domain is that of a mobile-manipulation robot. We know how to plan complex manipulations when we have exact models of the world, however even moderate amounts of uncertainty can cause the best laid plans to go astray. However, dealing with uncertainty is fundamental to manipulation since our knowledge of the world, whether through prior knowledge or sensors, is always limited. Our approach integrates planning, perceiving and acting to develop robot systems capable of autonomous mobile manipulation.

Results and Discussion

Our work encompasses several threads:

1. **Hierarchical planning in belief space**
2. **State estimation in complex spaces**
3. **Planning and control for manipulation**

The following sections summarize our research results in these areas. The work is reported in more detail in the accompanying papers.

1 Hierarchical planning in belief space

Unifying Perception, Estimation and Action for Mobile Manipulation via Belief Space Planning [9]

We have developed [9] an integrated strategy for planning, perception, state-estimation and action in complex mobile manipulation domains. The strategy is based on planning in the belief space of probability distribution over states. Our planning approach is based on hierarchical symbolic regression (pre-image back-chaining). We have developed a vocabulary of fluents that describe sets of belief states, which are goals and subgoals in the planning process. We have shown that a relatively small set of symbolic operators lead to task-oriented perception in support of the manipulation goals.

Figure 4 shows a sequence of images depicting the planning and execution process for an initial goal of placing the small blue cup at one end of the table. The robot starts with a known area around it, and the rest of the room is unknown, as represented in an oct-tree that maps the observed regions. To determine the contents of the swept regions of generated motions, a series of look motions. When these scans are executed (steps 2–7 in Figure 4), new areas of the oct-tree become known as illustrated in subsequent oct-trees in Figure 3. After the first two scans (steps 2 and 3 in Figure 4), the table has not been observed and so its pose distribution is diffuse – as shown in the first pose distribution in Figure 3. Also, the big red object has not been seen; note that it is not part of the initial model. After the table is scanned (in step 4 of Figure 4), its pose distribution becomes tight but the blue cup is still not visible (since it is occluded by the big red object), so its pose distribution is still diffuse, as shown in the second pose distribution of Figure 3. In the scan of step 4, the red object is also seen and added to the model (as seen in the third pose distribution of Figure 3). When going to look at the region (in step 5 of Figure 4), where the red object is to be moved, the robot serendipitously "sees" the blue cup and narrows its distribution, as seen in the fourth pose distribution of of Figure 3. If the blue cup had not been seen at this point, a plan would have been constructed to move the red object out of the way so as to enable looking at the blue cup. Steps 6 and 7 of Figure 4 are undertaken to ensure that the space that the robots needs to traverse while moving the red block and the blue cup are free of obstructions. After the required regions are known, the planning and execution proceeds as usual, resulting in a sequence of operations to move the red block to its target location, pick up the blue block and take it to its goal location (steps 8-12 of Figure 4).

Figure 3: The distribution of the objects relative to the mean robot pose.

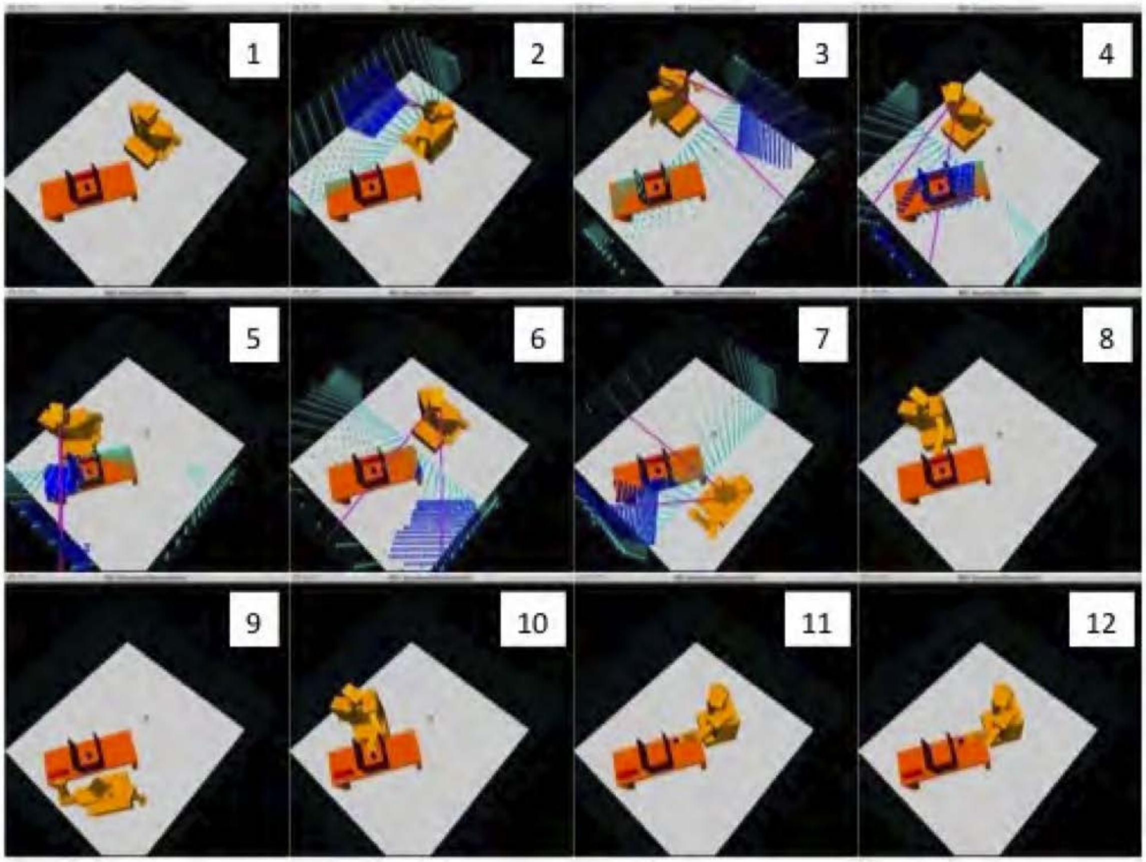


Figure 4: The key steps in the execution of a plan to place the small blue cup in a target region at one end of the table. The red object is initially not in the object's model of the world. Scans with the head-mounted sensor are shown as dark blue points. Scans with the scanning laser are shown in cyan.

Optimization in the Now: Dynamic Peephole Optimization for Hierarchical Planning [7]

For robots to effectively interact with the real world, they will need to perform complex tasks over long time horizons. This is a daunting challenge, but recent advances using hierarchical planning [9] have been able to provide leverage on this problem. Unfortunately, this approach makes no effort to account for the execution cost of an abstract plan and often arrives at poor quality plans.

We have developed a strategy for tackling the problem of optimization in hierarchical robotic planning [7] that addresses plan quality by dynamically reordering and grouping subgoals in an abstract plan. We reframe the cost estimation problem as one in which, given two subgoals G_1 and G_2, we must estimate which of the following strategies will be most efficient: planning for and executing G_1 first, planning for and executing G_2 first, or planning for them jointly and interleaving their execution (see Figure 5). Given the ability to answer that query, we will be able to perform "peephole optimization" of the plan at execution time, taking advantage of immediate knowledge of the current state of the world to select the best next action to take.

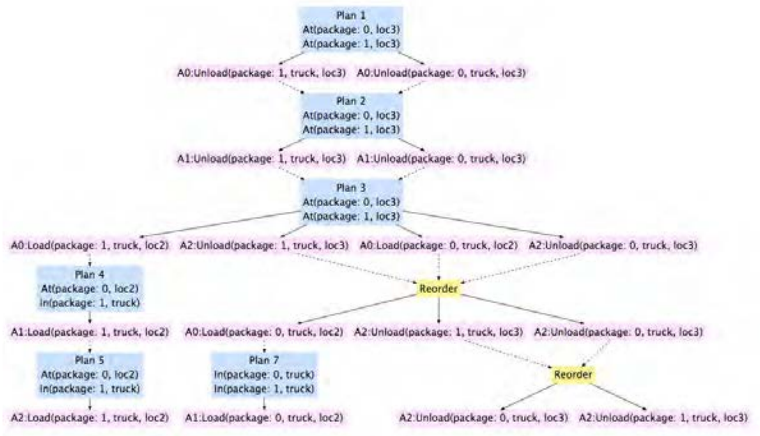


Figure 5: The root of a planning tree for a simple problem that involves transporting two packages to another location within the same city. At the high level, the Unload operators are recognized as overlapping on a shareable resource (truck) and are combined. In refining Plan 3, the Load operator is determined to overlap on both the shareable resource of the truck and the contained resource of the trucks location. It is reordered to be before the first Unload because it is estimated, greedily, as being easier to achieve from the current state. If there was not enough space in the truck, then the truck would not be considered shareable and the ordering would remain unchanged.

We ran experiments in challenging domains (Figure 6) and observed up to 30% reduction in execution cost when compared with a standard hierarchical planner (Figure 7).

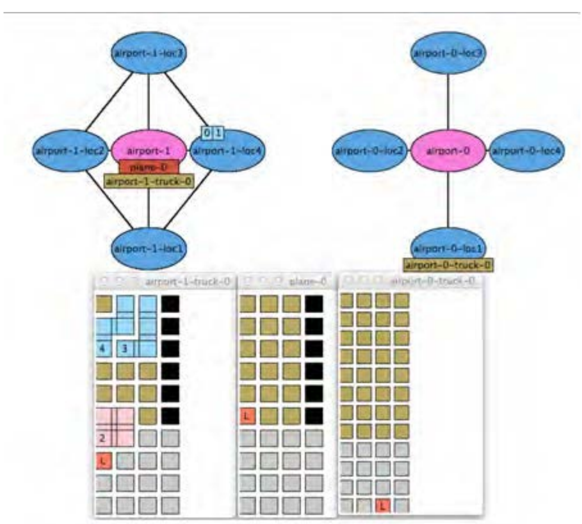


Figure 6: The Marsupial Logistics Domain. Circles are locations and pink circles are airports. The additional windows represent the loading and storage areas of the vehicles. The red squares represent a marsupial robot which takes care of storing packages for transit. In order for vehicles to move, all packages, as well as the loader, must be on one of the beige squares. Package 2 is about to be unloaded at airport-1 so it can be flown to a destination.

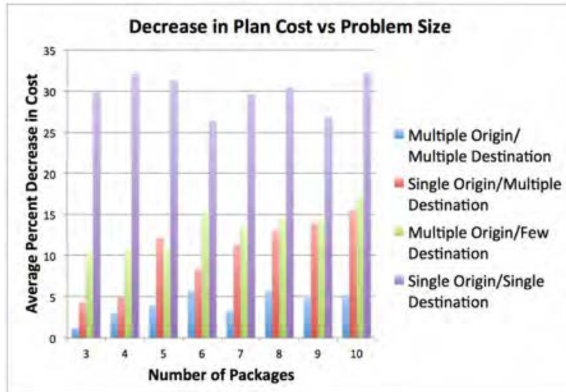


Figure 7: Average percent decrease in plan cost vs. problem size for RCHPN (with goal re-ordering) vs HPN (with no goal re-ordering).

Foresight and Reconsideration in Hierarchical Planning and Execution [14]

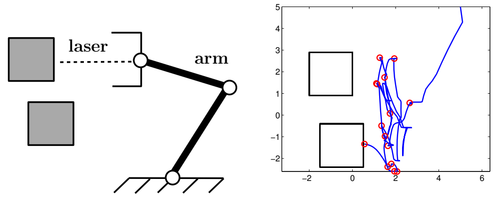
We have developed a hierarchical planning and execution architecture that maintains the computational efficiency of hierarchical decomposition while improving optimality. It provides mechanisms for monitoring the belief state during execution and performing selective replanning to repair poor choices and take advantage of new opportunities. It also provides mechanisms for looking ahead into future plans to avoid making short-sighted choices. The effectiveness of this architecture was shown through comparative experiments in simulation and demonstrated on a real PR2 robot navigating among (unknown) movable obstacles.

Non-Gaussian Belief Space Planning [17]

In partially observable control domains it is potentially necessary to perform complex information-gathering operations in order to identify the state. Our approach to solving these problems, as illustrated above, is to create plans in belief-space, the space of probability distributions over the underlying state of the system. The belief-space plan encodes a strategy for performing a task while gaining information as necessary. Unlike most approaches in the literature which rely upon representing belief state as a Gaussian distribution, we have developed [17] an approach to non-Gaussian belief space planning based on solving a non-linear optimization problem defined in terms of a (small) set of state samples.

We have shown that even though our approach makes optimistic assumptions about the content of future observations for planning purposes, all low-cost plans are guaranteed to gain information in a specific way under certain conditions. We have shown that eventually, the algorithm is guaranteed to localize the true state of the system and to reach a goal region with high probability. Although the computational complexity of the algorithm is dominated by the number of samples used to define the optimization problem, our convergence guarantee holds with as few as two samples. Moreover, we have shown empirically that it is unnecessary to use large numbers of samples in order to obtain good performance.

Figure 8(a) shows a simple application of the algorithm. A two-link robot arm moves a hand in the plane. A single range-finding laser is mounted at the center of the hand. The laser measures the range from the end-effector to whatever object it "sees". The hand and laser are constrained to remain horizontal. The position of the hand is assumed to be measured perfectly. There are two boxes of known size but unknown position to the left of the robot (four dimensions of unobserved state). The boxes are constrained to be aligned with the coordinate frame (they cannot rotate). The control input to the system is the planar velocity of the end-effector. The objective is for the robot to localize the two boxes using its laser and move the end-effector to a point directly in front of the right-most box (the box with the largest x-coordinate) so that it can grasp by extending and closing the gripper. Figure 8(b) shows a path found by the algorithm. The key point is that this path was found completely automatically; it is not an instance of a pre-programmed strategy.



(a) (b)

Figure 8: Illustration of non-Gaussian belief space planning using a nine-sample planner. The path found by the algorithm starts in the upper right and ends at a point directly in front of the right-most box. The red circles denote where re-planning occurred.

Integrated Task and Motion Planning in Belief Space [10]

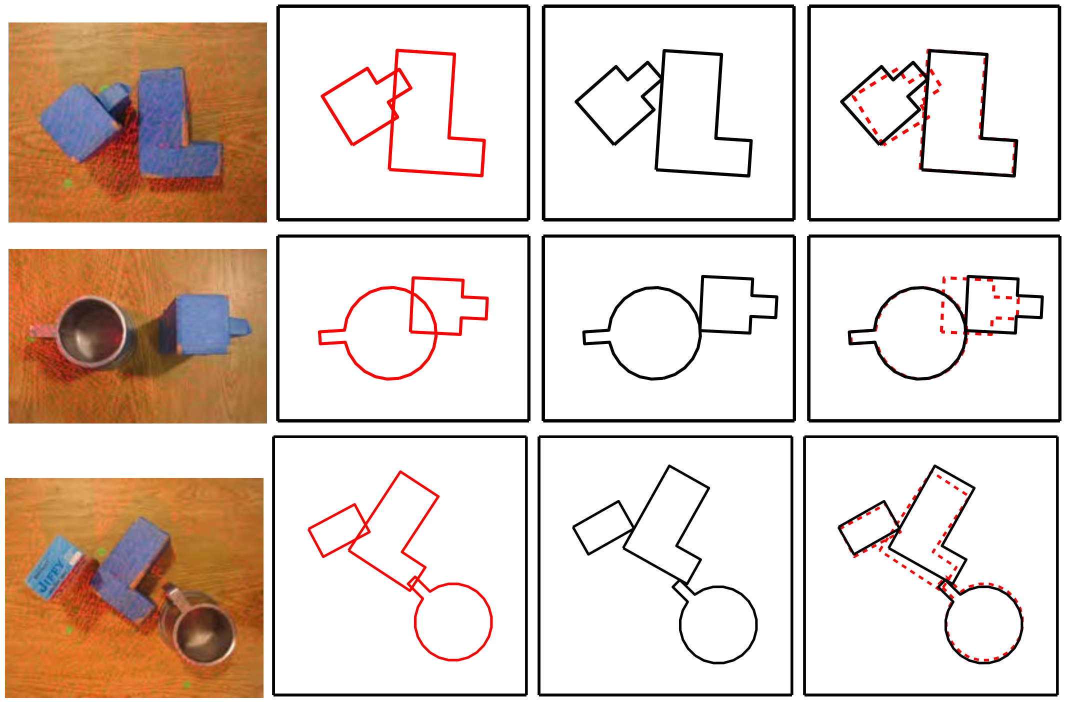
This is the journal-length description of our integrated strategy for planning, perception, state-estimation and action.

2 State estimation in complex spaces

Collision-Free State Estimation [18]

In state estimation, we often want the maximum likelihood estimate of the current state. For the commonly used joint multivariate Gaussian distribution over the state space, this can be efficiently found using a Kalman filter. However, in complex environments the state space is often highly constrained. For example, for objects within a refrigerator, they cannot interpenetrate each other or the refrigerator walls. The multivariate Gaussian is unconstrained over the state space and cannot incorporate these constraints. In particular, the state estimate returned by the unconstrained distribution may itself be infeasible.

We have developed [18] an approach that solves a constrained optimization problem (find poses with maximum probability subject to non-collision constraints) to find a good feasible state estimate. We have tested this for estimating collision-free configurations for objects resting stably on a 2-D surface and have demonstrated its utility in a real robot perception domain. Example of the results can be seen in Figure 9.



(a) Scene from above (b) Infeasible sample mean (c) Optimization solution (d) Superimposed image

Figure 9: Collision-Free Estimation

Manipulation-based Active Search for Occluded Objects [19]

Object search is an integral part of daily life, and in the quest for competent mobile manipulation robots it is an unavoidable problem. Previous approaches focus on cases where objects are in unknown rooms but lying out in the open, which transforms object search into active visual search. However, in real life, objects may be in the back of cupboards occluded by other objects, instead of conveniently on a table by themselves.

Extending search to occluded objects requires a more precise model and tighter integration with manipulation. We have developed [19] a novel generative model for representing container (e.g. cupboard, drawer, etc.) contents by using object co-occurrence information and spatial constraints.

To model object-object type similarity, we introduce the notion of a containers composition, a latent distribution over object types, with a prior based on co- occurrence statistics to enforce the known type similarities. Second, we enforce container spatial constraints by specifying a generative model for putting objects into containers, and then using it to sample contents of unobserved container regions. This generative process results in samples of container contents and configurations, which can be used to answer our fundamental query of object search: how likely is the target object to be found in a certain container?

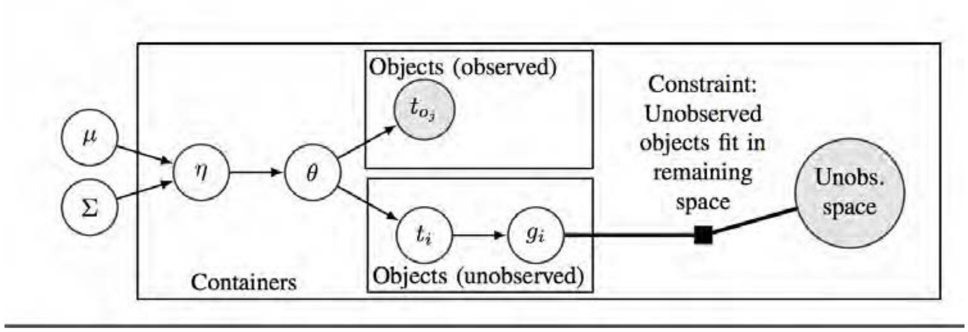


Figure 10: Graphical representation of our probabilistic model over container contents. Object types t are drawn independently from the composition θ. The prior on θ enforces object type similarities. During object search, parts of a container have been observed, with object types $\{t_{o_j}\}$ found. Unobserved objects with types $\{t_i\}$ may exist in the unobserved space; if so, they must all fit within. This spatial constraint is represented as a factor.

We have applied our model and the resulting search strategy for a mobile manipulator modeled on a Willow Garage PR2 robot. As shown in Figure 12, the robot is in an environment with 4 cupboards. Each cupboard has high sides and movable objects in the front that occlude the view of the rest of the contents. The robots goal is to locate the green cup, which in this example is in the back of cupboard N. Object type similarities are indicated by color, with green and brown objects tending to co-occur, and similarly for red and blue. The planning framework described in [9] is used.

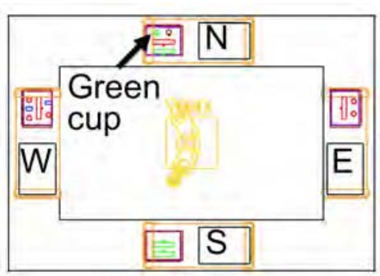


Figure 11

Figure 12 shows snapshots of the search. The top row is the robots belief state: gray areas show regions not yet viewed by the robot; colored objects show detected objects. The bottom row shows the estimate of $P(\mathrm{greencupinc}|\{\mathrm{t_{o_j}}\})$ for each container. From left to right:

(a) After seeing the front of each cupboard, object type similarity indicates only N and S are likely. Also, N is more likely because it has more unobserved space.

(b) When exploring N, an unexpected red object is observed. Since red objects tend not to co-occur with green objects, the probability in N drops, and S becomes more likely.

(c) Removing an object from S reveals that there is no more space behind the remaining object for a cup, so the probability becomes 0. Approaches that do not reason about spatial constraints would remove the remaining green object in S as well, since it is likely to co-occur with the target green cup.

(d) N is now the most likely container again. Removing the red object reveals the target green cup in the back of N.

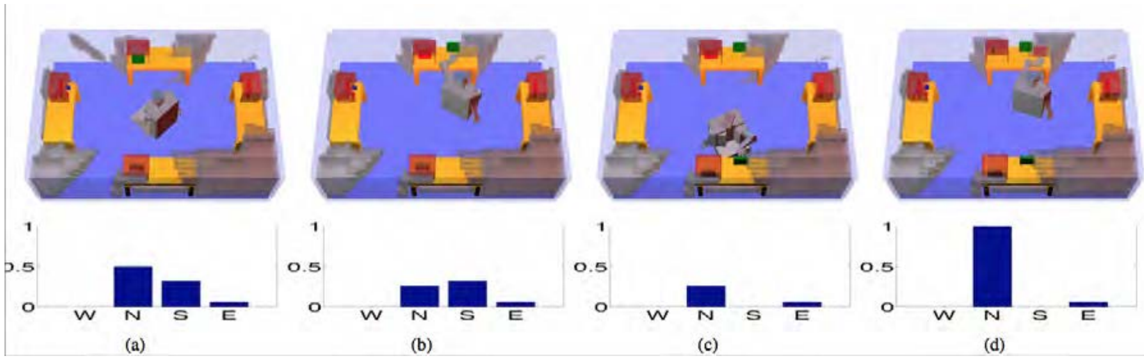


Figure 12

Interactive Bayesian Identification of Kinematic Mechanisms [1]

We have addressed the problem of identifying mechanisms based on data gathered while interacting with them – the robot tries to move a handle to various target locations and observes the reached location. Figure 13 shows Willow Garage PR2 robot manipulating an crank (described by a revolute model).

Figure 13: PR2 robot manipulating an crank (described by a revolute model)

We developed a decision-theoretic formulation of this problem, using Bayesian filtering techniques to maintain a distributional estimate of the mechanism type and parameters. In order to reduce the amount of interaction required to arrive at a confident identification, we select actions explicitly to reduce entropy in the current estimate.

We have demonstrated this approach on a domain with four primitive and two composite mechanisms. Diagrams of the each of the 6 models considered are shown in Figure **??**.

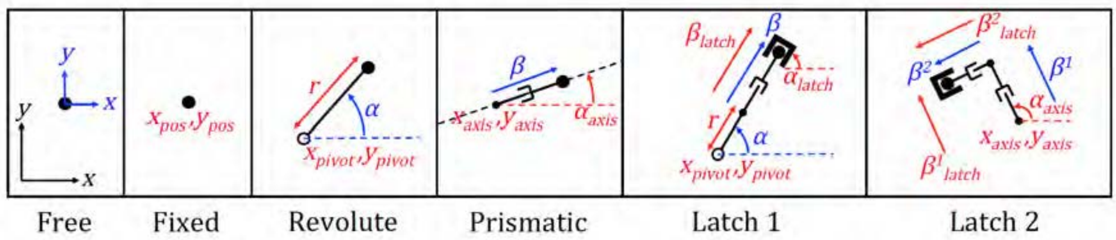
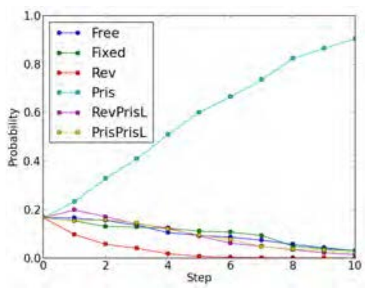
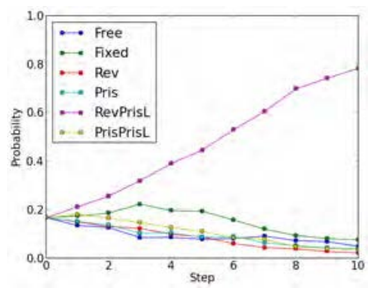


Figure 14: Mechanism models. Fixed parameters are shown in red while variables are shown in blue. The large dot represents each mechanisms handle.

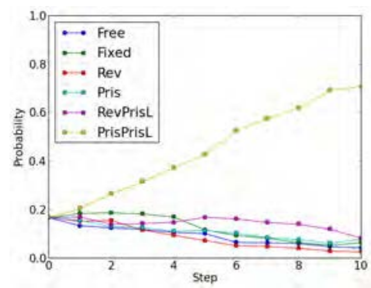
The results show that this approach can correctly identify complex mechanisms including mechanisms which are difficult to model analytically, such as, latches. The results also show that entropy-based action selection can significantly decrease the number of actions required to gather the same information. Figure 14 shows results for several mechanisms model type using several action selection schemes, including entropy-based selection.



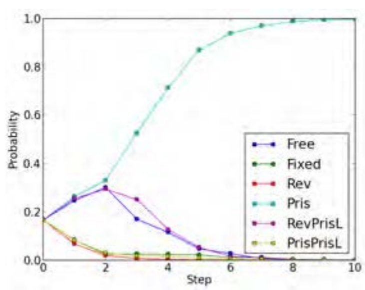
(a) Prismatic - Random



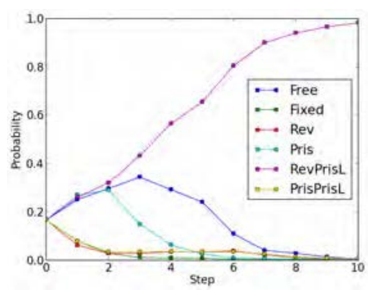
(b) Latch 1 - Random



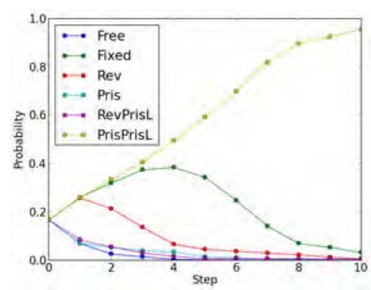
(c) Latch 2 - Random



(d) Prismatic - Entropy



(e) Latch 1 - Entropy

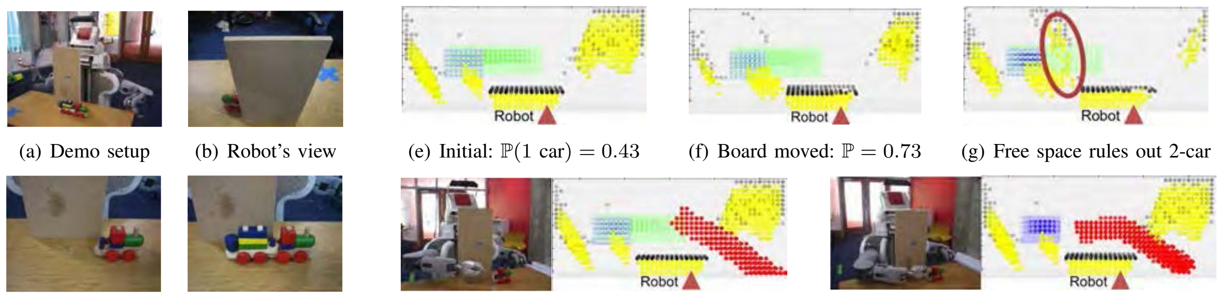


(f) Latch 2 - Entropy

Figure 15: Filter convergence and random vs. entropy-based action selection from Prismatic, Latch 1, and Latch 2 models. Each plot shows probability of mechanism types as a function of number of actions.

Not Seeing is Also Believing: Combining Object and Metric Spatial Information [20]

Spatial representations are fundamental to mobile robots operating in uncertain environments. Two frequently-used representations are occupancy grid maps, which only model metric information, and object-based world models, which only model object attributes. Many tasks represent space in just one of these two ways; however, because objects must be physically grounded in metric space, these two distinct layers of representation are fundamentally linked. We have developed an approach that maintains these two sources of spatial information separately, and combines them on demand. We illustrate the utility and necessity of combining such information through applying our approach to a collection of motivating examples.



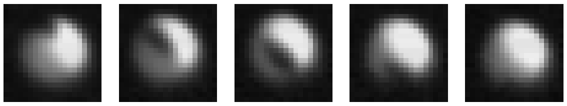
(a) Demo setup (b) Robot's view (e) Initial: $\mathbb{P}(1 \text{ car}) = 0.43$ (f) Board moved: $\mathbb{P} = 0.73$ (g) Free space rules out 2-car

(c) Is it 1 car? (d) Or 2 cars? (h) Arm moves inwards: $\mathbb{P}(1 \text{ car}) = 0.44$ (i) Arm overlaps and hence rules out 2-car case

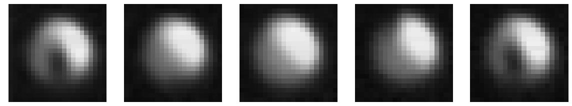
Figure 16: A 3-D demonstration on a PR2 robot. Plots show occupancy grids with 1m 0.4m 0.2m volume, containing 104 cubes of side length 2cm, with the final (vertical) dimension projected onto the table. Colors depict occupancy type/source: Yellow = free space observation; Black = occupancy observation; Blue = inferred occupancy from one-car train; Green = inferred occupancy from two-car train; Red = occupied by robot in its current state. In this projection, the robot is situated at the bottom center of the plot, facing upwards; the black line observed near the bottom corresponds to the board. (a)-(b) A toy train is on a table, but only part of the front is visible to the robot. (c)-(d) This is indicative of two possible scenarios: the train has one car or two cars; there is in fact only one car. (e)-(g) One way to determine the answer is to move the occluding board away. This reveals free space where the second car would have been (circled in (e)), hence ruling out the two-car case. (h)-(i) Another way is to use the robot arm. If the arm successfully sweeps through cells without detecting collision, the cells must have originally been free and are now occupied by the arm. Sweeping through where the second car would have been therefore eliminates the possibility of the train being there.

Tracking the Spin on a Ping Pong Ball with the Quaternion Bingham Filter [5]

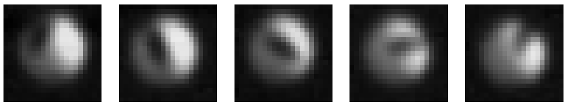
We have developed a deterministic method for sequential estimation of 3-D rotations. The Bingham distribution is used to represent uncertainty directly on the unit quaternion hypersphere. Quaternions avoid the degeneracies of other 3-D orientation representations, while the Bingham distribution allows tracking of large-error (high-entropy) rotational distributions. Experimental comparison to a leading EKF-based filtering approach on both synthetic signals and a ball-tracking dataset shows that the Quaternion Bingham Filter (QBF) has lower tracking error than the EKF, particularly when the state is highly dynamic. We present two versions of the QBF suitable for tracking the state of first- and second-order rotating dynamical systems.



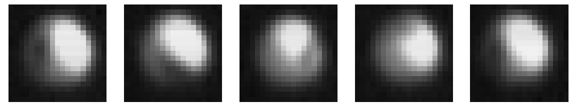
(a) slow top-spin



(b) fast top-spin



(c) slow side-spin



(d) fast side-spin

Figure 17: Example image sequences from the ping-pong ball dataset. In addition to lighting variations and low image resolution, high spin rates make this dataset extremely challenging for orientation tracking algorithms. Also, because the cameras were facing top-down towards the table, tracking side-spin relies on correctly estimating the orientation of the elliptical marking in the image, and is therefore much harder than tracking top-spin or under-spin.

Data association for semantic world modeling from partial views [21]

This is the journal-length description of our approach to data-association for state-estimation in complex domains.

3 Planning and control for manipulation

LQR-RRT*: Optimal Sampling-Based Motion Planning with Automatically Derived Extension Heuristics [16]

A key insight of our work has been that planning in belief space is an instance of the general problem of planning for underactuated systems. Essentially, we have only indirect control on the uncertainty through the dynamics in the belief space. One popular approach to planning for underactuated systems is the RRT (Rapidly-exploring Random Trees) algorithm. However, the RRT gives wildly unoptimal results.

The RRT* algorithm has recently been proposed as an optimal extension to the standard RRT algorithm. However, like RRT, RRT* is difficult to apply in problems with complicated or underactuated dynamics because it requires the design of a two domain-specific extension heuristics: a distance metric and node extension method.

We have developed [16] a method to automatically deriving these two heuristics for RRT* by locally linearizing the domain dynamics and applying linear quadratic regulation (LQR). The resulting algorithm, LQR-RRT*, finds optimal plans in domains with complex or underactuated dynamics without requiring domain-specific design choices.

Figure 17(b) shows the Light-Dark domain, a partially observable problem where the agent must move into a goal region with high confidence. Initially, the agent is uncertain of its true position. On each time step, the agent makes noisy state measurements (less noisy in the bright areas). Since the agent is unable to sense state directly, it is instead necessary to plan in the space of beliefs regarding the underlying state of the system rather than the underlying state itself. In this example, it is assumed that belief state is always an isotropic Gaussian such that belief state is three-dimensional: two dimensions describe the mean and one dimension describes variance.

The goal of planning is to move from an initial high-variance belief state to a low-variance belief state where the mean of the distribution is at the goal. This objective corresponds to a situation where the agent is highly confident that the true state is in the goal region. In order to achieve this, the agent must move toward one of the lights in order to obtain a good estimate of its position before proceedings to the goal. The domain, along with a sample solution trajectory, is depicted in Figure 17.

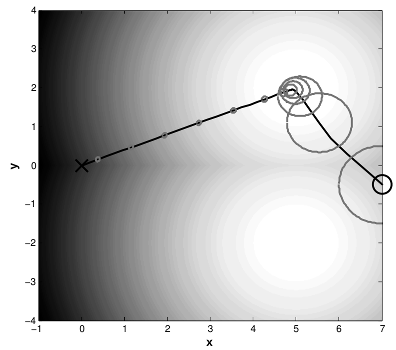


Figure 18: The Light-Dark domain, where the noise in the agent's location sensing depends upon the amount of light present at its location. Here the agent moves from its start location (marked by a black circle) to its goal (a black X), first passing through a well-lit area to reduce its localization uncertainty (variance shown using gray circles).

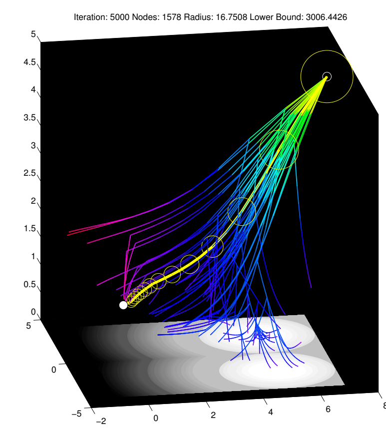


Figure 19: The search tree for Light-Dark after 5000 iterations of planning. The belief space is represented in 3-dimensions, where the mean of the agent's location estimate is the "floor", and the vertical axis represents the variance of the agent's belief distribution; lower points represent less location uncertainty. The algorithm builds a search tree to find a trajectory through belief-space. The optimized solution is shown as a thick yellow line, where the agent moves towards the goal while lowering the variance of its location distribution (descending the vertical axis). Policies in the tree are false-colored (green through magenta) as cost increases.

A Hierarchical Approach to Manipulation with Diverse Actions [2]

Most motion planning has focused on moving without contacts (collisions). However, for mobile manipulation, we face more general problems. We are given a mobile robot, a set of movable objects, and a set of diverse, possibly non-prehensile manipulation actions, such as Push, and the goal is to find a sequence of actions that moves each of the objects to a goal configuration. We call these Diverse Action Manipulation (DAMA) problems [2].

Our algorithm, DARRT, the Diverse Action Rapidly Exploring Random Tree, has the structure of a rapidly exploring random tree (RRT) with controls. However, we modify both the extension and sampling methods to work with manipulation. In particular, we have found that we need to sample from various projections into subsets of the full joint configuration space (of the robot and the objects being manipulated) so as to avoid the danger of needing to sample from measure zero subsets of the configuration space. We have also shown that the DAMA problem can be framed as a multi-modal planning problem and developed a hierarchical algorithm that takes advantage of this multi-modal nature.

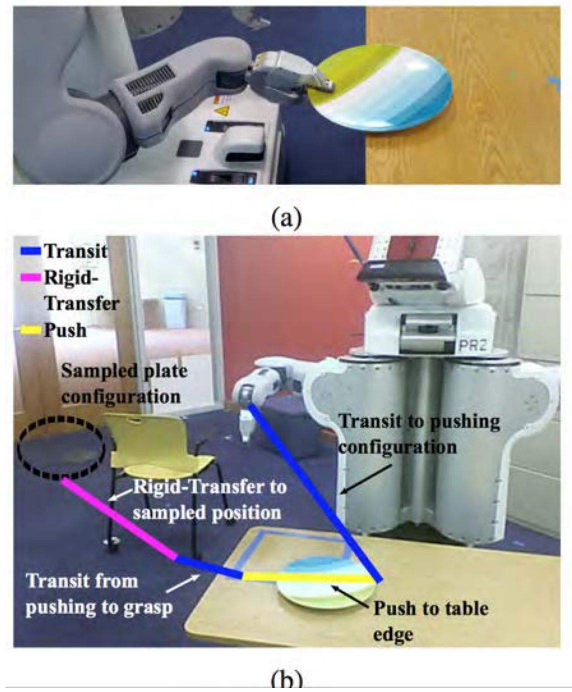


Figure 20: An example world with Transit, Rigid-Transfer and Push primitives.

An example world with Transit, Rigid-Transfer and Push primitives in shown in Figure 19.

(a) If the robot can only grasp the plate when it is at a single point on the edge of the table, this is a zero-measure subset of the configurations in which the plate is on the table.

(b) An extension from the state shown in the photograph towards the sample shown with the dashed lines. Samples specify only partial states; here the sample specifies a configuration for the plate. The sequence of primitives shown for the wrist first transits the robot to a pushing configuration (blue), pushes the plate towards the edge of the table (yellow), transits the robot to a grasp (blue), and finally transfers the plate to its sampled configuration (magenta).

We have tested our algorithms in complicated manipulation domains, as shown in Figure 20.

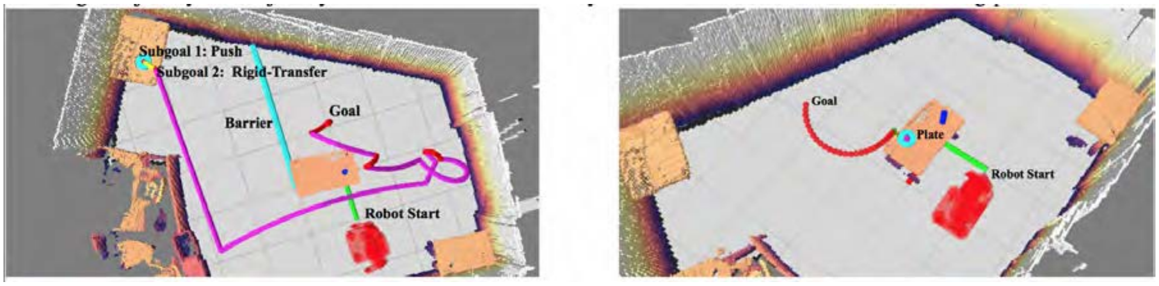


Figure 21: There is a plate (cyan cylinder) on a table and the goal is to move it to somewhere else in the environment. The robot starting state is shown in red and the final trajectory is shown color-coded by the primitive used. The trajectory is only shown for the plate for visual clarity, but the plans were for robot and object.

We have demonstrated that the new algorithms are effective in complex domains, and that the hierarchical algorithm is usually much more efficient than the forward or bi-directional searches.

Object Placement as Inverse Motion Planning[8]

There has been little systematic investigation of robust manipulation primitives. Consider placing, a very common operation in manipulation, it has many failure modes, as shown in Figure 21, and no general methods exist for planning robust placing.

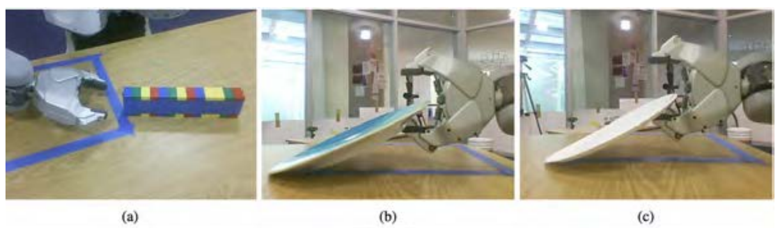


Figure 22: Placing failure modes. Failure modes for the different objects. (a) The tower could tip over. (b)-(c) The plates got caught on the gripper and dragged during retreat.

We have been investigating how to use the environment, including the robots other hand, to constrain the possible motions of an object during placement [8]. This problem is an instance of the inverse motion planning problem, in which we solve for a configuration of the environment that makes desired trajectories likely.

To calculate the probability that an object will take a particular trajectory, we model the physics of placing as a mixture model of simple object motions. Our algorithm searches over the possible configurations of the object and environment and uses this model to choose the configuration most likely to lead to a successful place (Figure 22). We show that this algorithm allows the PR2 robot to execute placements that fail with traditional placing implementations (Figure 23).

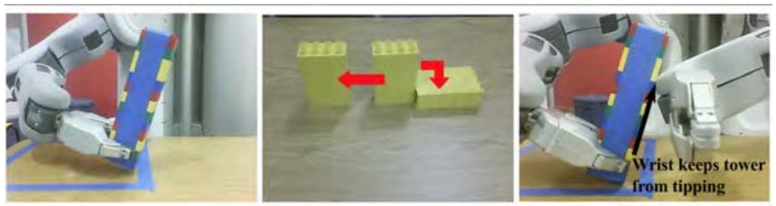


Figure 23: A simple algorithm for robust placing. (a) We first search over release configurations for the configuration with the highest probability of success. (b) Given the release configuration, we evaluate the outcomes of the failure modes. Tipping causes a rotation around the x axis while dragging causes a translation along the y axis as shown by the red arrows. We order these by decreasing probability. (c) Given the release configuration r found in step (a) and the order of the failure mode outcomes found in step (b), we use the movable objects - in this case, the robots empty gripper - to block the most likely failure modes.

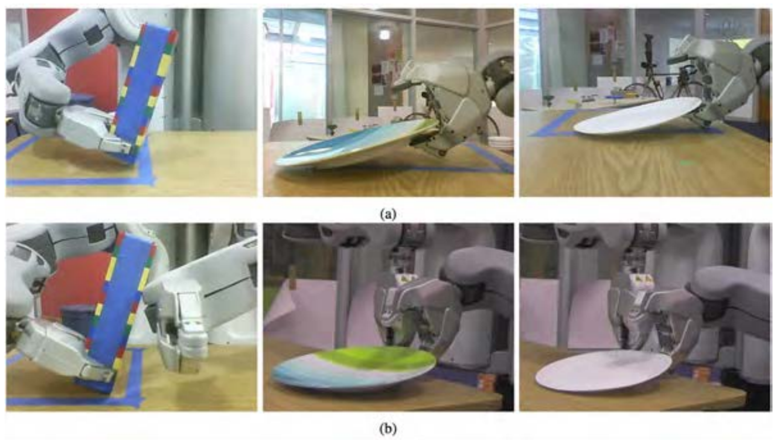


Figure 24: Example release configurations. (a) Passive placing results. (b) Robust placing results.

Optimal Sampling-Based Planning for Linear-Quadratic Kinodynamic Systems [6]

A key insight of our work has been that planning in belief space (the space of probability distributions over the underlying states) is an instance of the general problem of planning for underactuated systems. Essentially, we have only indirect control on the uncertainty through the dynamics in the belief space. One popular approach to planning for underactuated systems is the RRT (Rapidly-exploring Random Trees) algorithm. However, the RRT gives wildly unoptimal results.

The RRT* algorithm has recently been proposed as an optimal extension to the standard RRT algorithm. However, like RRT, RRT* is difficult to apply in problems with complicated or underactuated dynamics because it requires the design of a two domain-specific extension heuristics: a distance metric and node extension method.

We have developed a new approach [6] to applying LQR to the problem of finding optimal finite-horizon extension trajectories and costs in the context of RRT. This new algorithm converges, with probability one, to the optimal plan for problems with affine dynamics and quadratic cost functions. We include time as an additional dimension of the space in which the tree grows, an approach commonly used to solve problems in time-varying environments

Because the search tree explicitly represents state-time and explores all possible trajectories in this space, we can set contraints on the length of time of the solutions. This makes the algorithm applicable to a wider range of problems. In particular, we show that for any linear dynamical system with a quadratic cost function, our algorithm guarantees the probabilistic optimality of the resulting trajectory. Moreover, the algorithm can be directly extended to non-linear systems by linearizing the dynamics at vertices in the tree. These approximate dynamics are, in general, affine, i.e., containing a zero-order term. LQR is typically applied to linear systems, so we also include an extension to LQR which can be applied to affine systems. Experimental results (see Figure 24 and Figure 25) suggest that our algorithm obtains good results in these settings.

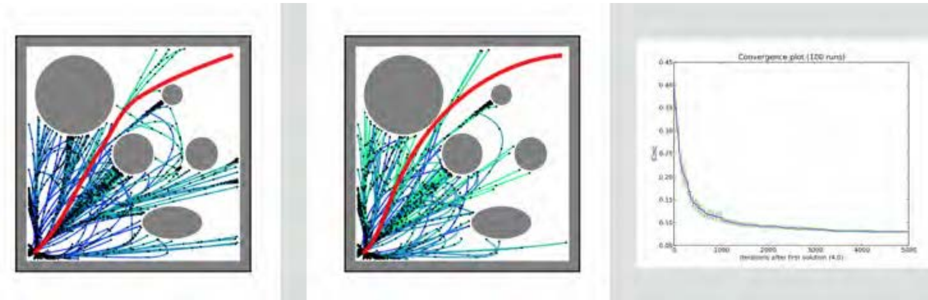


Figure 25: Solution tree generated by our algorithm while solving the double integrator problem. (a) The tree is grown in the domain. The system must move from a stationary position in the lower left to a stationary position in the upper right. The five ellipses denote obstacles, and the tree is color-coded for cost. The thick red line shows the current best trajectory. (b) All candidate solution trajectories found during 100 separate runs of the algorithm, again color coded for cost.

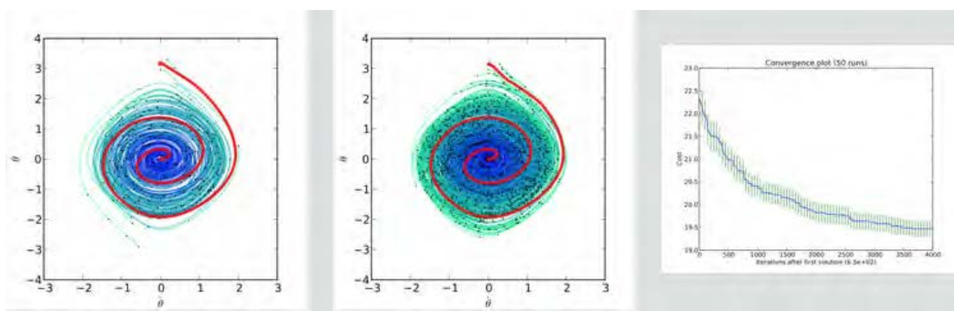


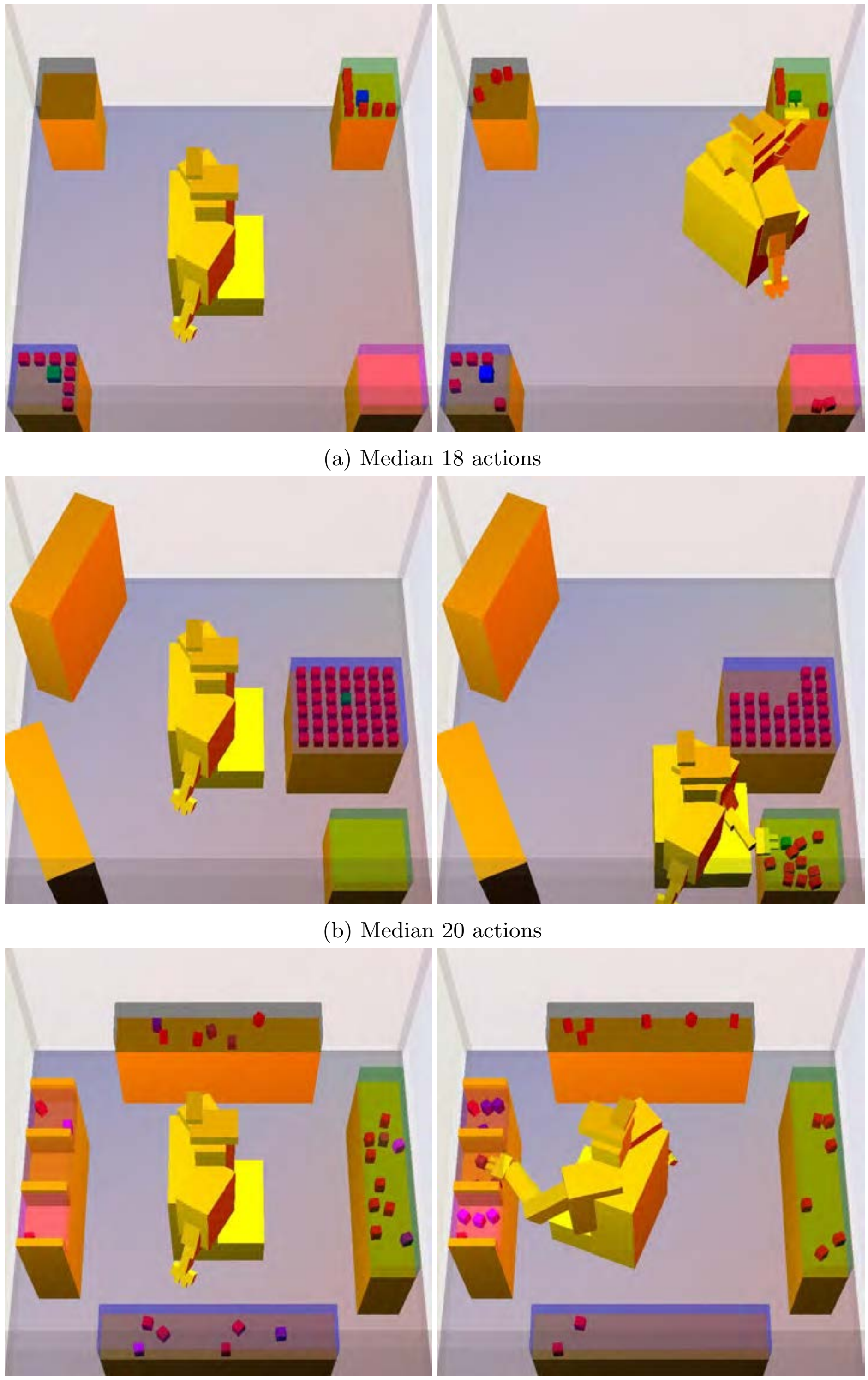
Figure 26: Our algorithm applied to the inverted pendulum. (a) through (c) illustrate a phase plot of the RRT tree after 500, 1000, and 1500 iterations, respectively. The red line in each plot shows the lowest cost solution in the tree (after 500 iterations, no solution has been found). Paths are colored according to cost (from dark blue to light cyan). (d) shows performance averaged over 50 runs (average in blue; standard error bars in green.)

FFRob: An efficient heuristic for task and motion planning [3]

We considered manipulation problems involving many objects. These problem present substantial challenges for motion planning algorithms due to the high dimensionality and multi-modality of the search space. Symbolic task planners can efficiently construct plans involving many entities but cannot incorporate the constraints from geometry and kinematics.

We have show how to extend the heuristic ideas from one of the most successful symbolic planners in recent years, the FastForward (FF) planner, to motion planning, and to compute it efficiently. We use a multi-query roadmap structure that can be conditionalized to model different placements of movable objects. The resulting tightly integrated planner is simple and performs efficiently in a collection of tasks involving manipulation of many objects.

We tested our algorithm on 6 different tasks, in which the goals were conjunctions of $In(O_i, Rj)$ for some subset of the objects (the ones not colored red). Other objects were moved as necessary to achieve these goals. The last three tasks are shown in Figure 26; the first three are tasks are simpler variations on task 3 (Figure 26(a)).



(a) Median 18 actions

(b) Median 20 actions

(c) Median 32 actions

Figure 27: The initial and final state in three of the tasks in the experiments.

A constraint-based method for solving sequential manipulation planning problems [15]

We have developed a strategy for integrated task and motion planning based on performing a symbolic search for a sequence of high-level operations, such as pick, move and place, while postponing geometric decisions. Partial plans (skeletons) in this search thus pose a geometric constraint-satisfaction problem (CSP), involving sequences of placements and paths for the robot, and grasps and locations of objects. We develop a formulation for these problems in a discretized configuration space for the robot. The resulting problems can be solved using existing methods for discrete CSP.

Figure 28: Plan to pick an object from an awkward location on one table, and move it to the back table.

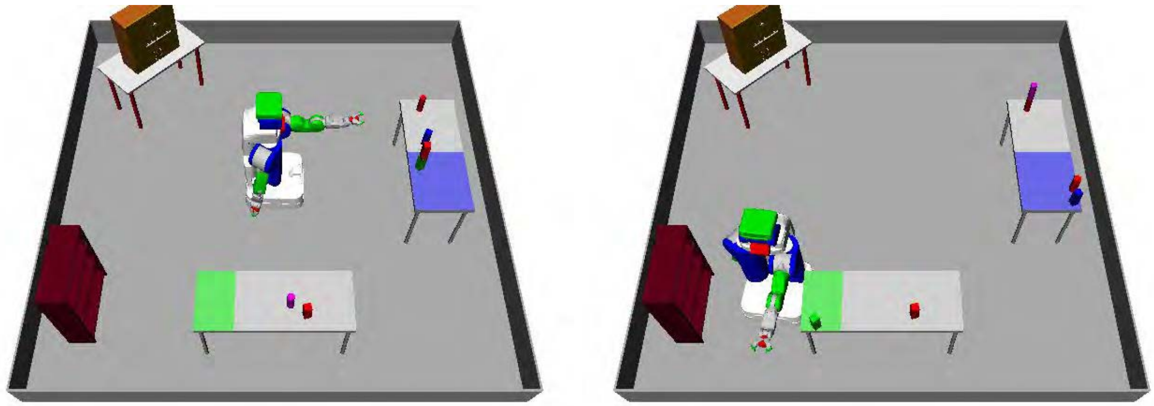
Backward-Forward Search for Manipulation Planning [4]

We are interested in solving manipulation planning problems in high-dimensional hybrid configuration spaces. A state of such a system is characterized by a finite set of configuration variables that may be discrete (such as which object a robot is holding or whether the light is turned on) or continuous (such as the joint-space configuration of a robot or the pose of an object).

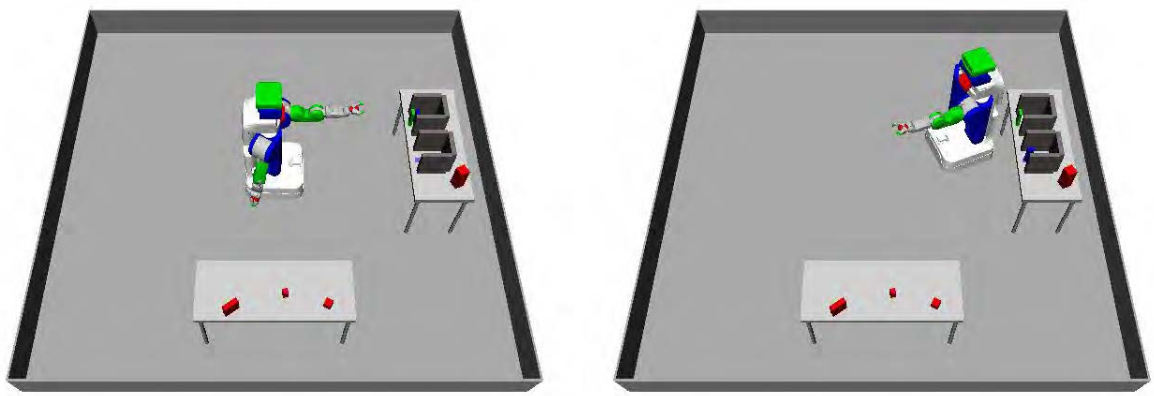
Without making any assumptions about the nature of the configuration space and the transition dynamics, planning in such a space is quite difficult. We have developed a problem representation that can reveal useful underlying structure in the domain that will be exploited by our method. There are three important kinds of leverage:

- *Factoring and sparsity*: by representing the state space as the product of the spaces of a set of state variables, we are able to assert that each action of the robot affects only a small subset of the state variables, allowing individual actions to be contemplated in state spaces that are effectively much smaller.
- *Continuous modes*: there are some continuous subspaces of the whole space that have continuous dynamics, which allows us to use classic sample-based robot motion planning techniques to move within those subspaces.
- *Heuristic estimates*: by constructing relaxed versions of a planning problem, we can efficiently obtain estimates of the cost to reach a goal state and use these estimates to make the search for a solution much more efficient.

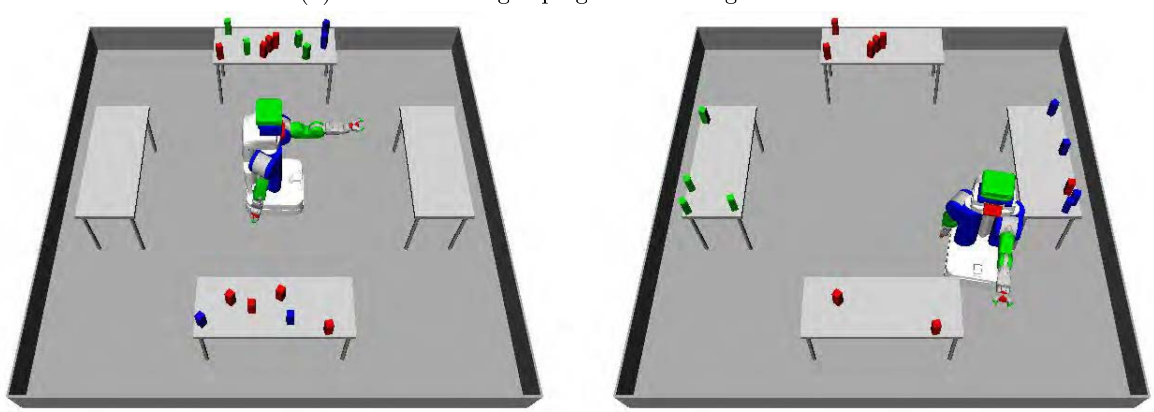
We have developed a new planning algorithm, HHBF, and applied it to three different manipulation problems (shown below) to characterize its performance. Solving these problems requires stacking, regrasping, and long-horizon manipulation. The planner and PR2 robot manipulation simulations were written in Python using OpenRAVE. In each problem, red objects represent moveable objects that have no particular goal condition. However, they impose geometric constraints on the problem and must, in many cases, be manipulated in order to produce a satisfying plan. For example, in problem 1, a stacked red block prevents the green block from being clear, so the planner first plans to unstack it before moving the green block. Problem 3 is designed to be comparable to trial 6 of the FFRob planning system [3].



(a) Problem 1: Moving blue block, clearing green block, and stacking purple cylinder



(b) Problem 2: Regrasping of blue and green blocks



(c) Problem 3: Sorting blue and green blocks task similar to [3]

Figure 29: The initial and final state for each problem.

Hierarchical planning for multi-contact non-prehensile manipulation [13]

We have explored a hierarchical approach to planning sequences of non-prehensile and prehensile actions. Our planner operates hierarchically, first finding a sequence of qualitative "object contact states" that characterize which parts of the moving object are in contact with which parts of other objects, then finding a feasible sequence of poses for the object (figure 29), and finally finding a sequence of contact points for the manipulators on the object (figure 30). This hierarchical structure provides significant search guidance, and divides the problem into three search problems that are much smaller than a search in the full combined configuration space of the object and manipulators.

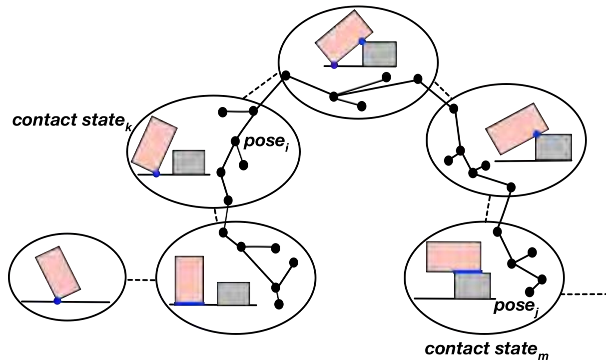


Figure 30: A contact state graph with poses connected through linear interpolation. Poses connecting two contact states are very close to each other.

To find a robot-contact plan, we discretize the object's surface into a set of possible contact points and define a state to contain an object pose and a set of contacts of the robot's manipulators on the object. We then identify states that are *feasible:* both *accessible*, meaning that the robot can reach all of the specified contacts and *stabilizable*, meaning that there exists a set of contact forces between the object and the robot's manipulators, as well as the fixed objects, that can stabilize the object against gravity (figure 30).

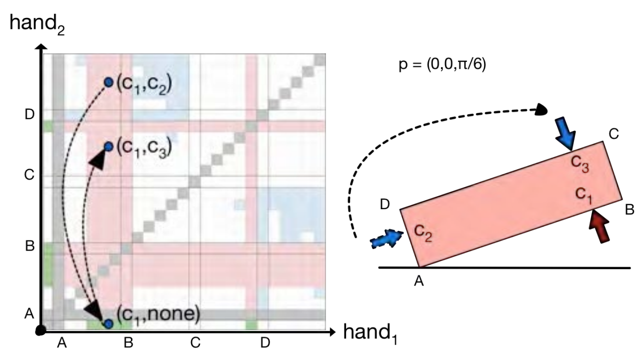


Figure 31: Robot contact space for $p = (0, 0, \pi/6)$. Each axis represents possible contact points along the object's surface accessible by $hand_1$ and $hand_2$. The leftmost column and the bottom row represent no-contact for $hand_1$ and $hand_2$, respectively. Green cells represent feasible states with only one contact, i.e. where the object can be balanced by only one hand. If either hand makes the object stabilizable on its own, the other hand can place itself on any accessible surface; these states are colored in red. For example, if a row's leftmost cell is green, all accessible cells in the row becomes red. States that require both hands are colored in blue. Grey cells represent invalid or inaccessible states. Since vertex A is already in contact with ground, any state containing A is inaccessible. White cells are infeasible. A *transit* is a transition from a red state to another red state in the same row or column. The example shows transits from (c_1, c_2) to $(c_1, none)$ to (c_1, c_3), changing which manipulator is stabilizing the object.

Figure 31 illustrates the connected search spaces: within the discrete contact states in the contact-state graph, there are individual object poses, and a path through object-contact space can be realized by a path through object pose space. Then, for each object pose, there is a set of robot contacts, and a path through object pose space can be realized by a path of transit and transfer motions through the combined space of robot contacts and object poses.

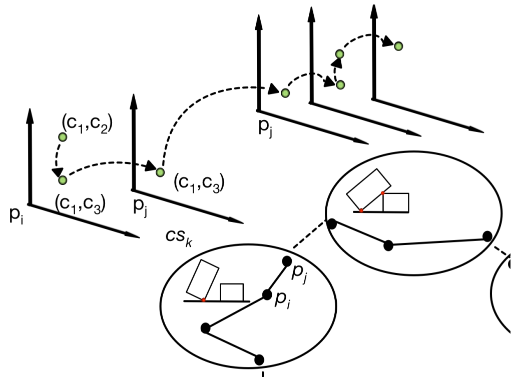


Figure 32: The relationship between the spaces of object contacts, object poses, and robot contacts.

We have implemented a version of this planner (in simulation) for planar objects and two robot contacts, without any further kinematic or collision constraints introduced to model the robot performing the manipulation. We tested these approaches on two problems. The first, shown in figure 32, focuses on sequencing non-prehensile manipulation steps. There is an obstacle in the middle of the table, and the goal in this problem is to move the box to the other side of the table. Allowing only nonprehensile manipulation, the planner is able to find a solution.

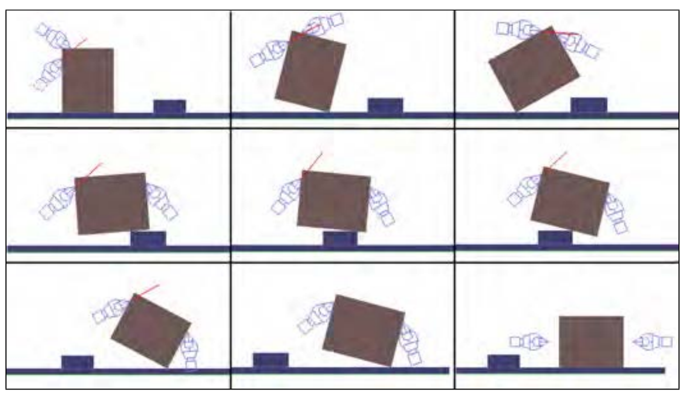


Figure 33: Key frames from a sample solution trajectory for tumbling one box over another. Red lines indicate the force direction of the robot contact pushing the object. The "hands" simply highlight the location of the chosen robot contacts.

Generalizing over Uncertain Dynamics for Online Trajectory Generation [11]

Given a known deterministic model of the dynamics of a system, a start and goal state, and a cost function to be minimized, trajectory optimization methods can be used to generate a trajectory that connects the start and goal states, respects the constraints imposed by the dynamics, and (locally) minimizes the cost subject to those constraints. A significant limitation to the application of these methods is the computational time required to solve the difficult non-linear program required to generate a near-optimal trajectory. In addition, standard techniques require the transition dynamics to be known with certainty.

We are interested in solving problems online in domains that are not completely understood in advance and that require efficient action selection. In such domains we will not know, offline, the exact dynamics of the system we want to control. Online, we will receive information that results in a posterior distribution over the domain dynamics. We seek to design an overall method that combines *offline* trajectory optimization and inductive learning methods to construct an *online* execution system that efficiently generates actions based on observations of the domain.

More concretely, we aim to build a trajectory generator that, for a given initial state and goal, maps the values of the dynamics parameters to a trajectory in the observable setting, or maps from an observation to a trajectory in the partially observable setting. We do this by training a regression function that maps both the dynamics parameters and the current system state to an appropriate control action. The trajectories used for off-line training are generated by using an existing trajectory optimizer that solves non-linear programs. To minimize the number of training trajectories required, we take an active-learning approach that uses an anomaly-detection strategy to determine which parts of the space require additional training data.

We considered two general problem settings. In the *completely observable* setting, we assume that at execution time the world dynamics will be fully observed; in the manipulation domain, this would correspond to observing the friction and COM of the object. In the *partially observable* setting, we assume that properties of the domain that govern its dynamics are only partially observed; for example, observing the height and shape of an object would allow us to make a "guess" in the form of a posterior distribution over these parameters to the dynamics, conditioned on the online observations. In both cases, we desire the online action-selection to run much more quickly than would be possible if it were necessary to run a traditional trajectory optimization algorithm online.

In one domain we have considered, the task is to move a cylindrical object with a multi-fingered robot hand from an initial position to a goal position. We find that when the system dynamics are observable, TOIL selects appropriate pushing trajectories, but when they are only partially observable, TOIL makes more robust choices; an example is illustrated in Figure 33.

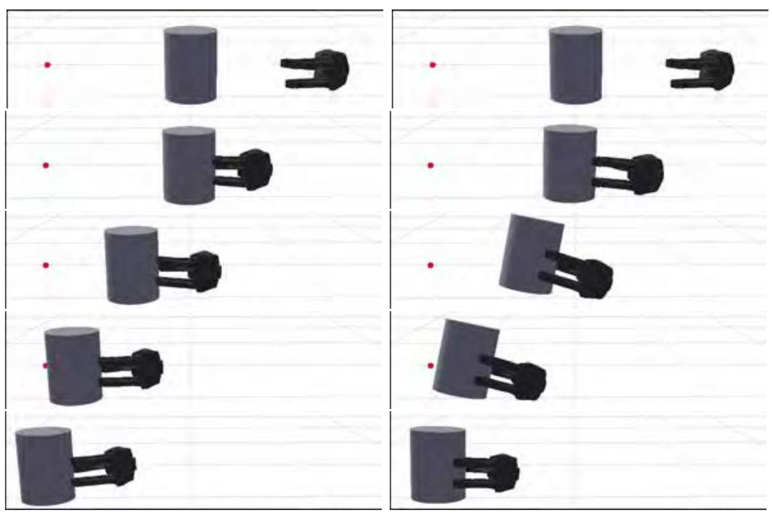


Figure 34: Trajectories for the observable (left) and partially observable case (right). For the observable case, the robot simply pushes the object to the goal. For the partially observable case, the robot lifts the object to to the goal, as to minimize the risk of tipping the cylinder over.

Symbol Acquisition for Probabilistic High-Level Planning [12]

Systems that combine high-level planning with low-level control are capable of generating complex, goal-driven behavior. But, they are hard to design because they require a difficult integration of symbolic reasoning and low-level motor control.

Recently, we showed how to automatically construct a symbolic representation suitable for planning in a high-dimensional, continuous domain. This work modeled the low-level domain as a semi-Markov decision process and formalized a propositional symbol as the name given to a grounding set of low-level states (represented compactly using a learned classifier). Their key result was that the symbols required to determine the feasibility of a plan are directly determined by characteristics of the actions available to an agent. This close relationship removes the need to hand-design symbolic representations of the world and enables an agent to, in principle, acquire them autonomously.

However, a set-based symbol formulation cannot deal with learned sets that may not be exactly correct, and can only determine whether or not the probability of successfully executing a plan is 1. These restrictions are ill-suited to the real-world, where learning necessarily results in uncertainty and all plans have some probability of failure

In our new work, we introduced a probabilistic reformulation of symbolic representations capable of naturally dealing with uncertain representations and probabilistic plans. This is achieved by moving from sets and logical operations to probability distributions and probabilistic operations. We use this framework to design an agent that autonomously learns a completely symbolic representation of a computer game domain, enabling very fast planning using an off-the-shelf probabilistic planner.

References

[1] Patrick R. Barragán, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Interactive bayesian identification of kinematic mechanisms. In *IEEE Conference on Robotics and Automation (ICRA)*, 2014.

[2] Jennifer Barry, Leslie Pack Kaelbling, and Tomas Lozano-Perez. A hierarchical approach to manipulation with diverse actions. In *IEEE Conference on Robotics and Automation (ICRA)*, 2013.

[3] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Ffrob: An efficient heuristic for task and motion planning. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2014.

[4] Caelan Reed Garrett, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Backwar-forward search for manipulation planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[5] Jared Glover and Leslie Pack Kaelbling. Tracking the spin on a ping pong ball with the quaternion bingham filter. In *IEEE Conference on Robotics and Automation (ICRA)*, 2014.

[6] G. Goretkin, A. Perez, R. Platt, and G.D. Konidaris. Optimal sampling-based planning for linear-quadratic kinodynamic systems. In *IEEE Conference on Robotics and Automation (ICRA)*, 2013.

[7] Dylan Hadfield-Menell, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Optimization in the now: Dynamic peephole optimization for hierarchical planning. In *IEEE Conference on Robotics and Automation (ICRA)*, 2013.

[8] Anne Holladay, Jennifer Barry, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Object placement as inverse motion planning. In *IEEE Conference on Robotics and Automation (ICRA)*, 2013.

[9] Leslie Pack Kaelbling and Tomas Lozano-Perez. Unifying perception, estimation and action for mobile manipulation via belief space planning. In *IEEE Conference on Robotics and Automation (ICRA)*, 2012.

[10] Leslie Pack Kaelbling and Tomas Lozano-Perez. Integrated task and motion planning in belief space. *International Journal of Robotics Research*, 2013.

[11] Beomjoon Kim, Albert Kim, Hongkai Dai, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Generalizing over uncertain dynamics for online trajectory generation. In *International Symposium of Robotics Research*, 2015.

[12] G.D. Konidaris, L.P. Kaelbling, and T. Lozano-Perez. Symbol acquisition for probabilistic high-level planning. In *Proceedings of the Twenty Fourth International Joint Conference on Artificial Intelligence*, 2015.

[13] Gilwoo Lee, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Hierarchical planning for multi-contact non-prehensile manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[14] Martin Levihn, Leslie Pack Kaelbling, Tomás Lozano-Pérez, and Mike Stilman. Foresight and reconsideration in hierarchical planning and execution. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[15] Tomas Lozano-Perez and Leslie Pack Kaelbling. A constraint-based method for solving sequential manipulation planning problems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[16] Alejandro Perez, Robert Platt Jr., George Konidaris, Leslie Kaelbling, and Tomas Lozano-Perez. LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. In *IEEE Conference on Robotics and Automation (ICRA)*, 2012.

[17] Robert Platt, Leslie Kaelbling, Tomas Lozano-Perez, and Russ Tedrake. Non-gaussian belief space planning: Correctness and complexity. In *IEEE Conference on Robotics and Automation (ICRA)*, 2012.

[18] Lawson L. S. Wong, Leslie P. Kaelbling, and Tomas Lozano-Perez. Collision-free state estimation. In *IEEE Conference on Robotics and Automation (ICRA)*, 2012.

[19] Lawson L. S. Wong, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Manipulation-based active search for occluded objects. In *IEEE Conference on Robotics and Automation (ICRA)*, 2013.

[20] Lawson L.S. Wong, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Not seeing is also believing: Combining object and metric spatial information. In *IEEE Conference on Robotics and Automation (ICRA)*, 2014.

[21] Lawson L.S. Wong, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Data association for semantic world modeling from partial views. *International Journal of Robotics Research*, 34(7), 2015.